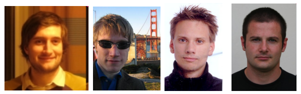
Real-Time Optimization of Large Distributed Systems

Moritz Diehl
Optimization in Engineering Center (OPTEC) & Electrical Engineering Department
KU Leuven, Belgium

joint work with
Attila Kozma, Hans Joachim Ferreau, Joel Andersson, and Carlo Savorgnan



Overview

- Large Interconnected Systems
- Distributed Multiple Shooting
- Outlook: Distributed Quadratic Programming

Motivation

Large scale systems in engineering are

- composed of **multiple subsystems**
- each with complex **nonlinear dynamics**
- and coupled by **mutual interactions**

chemical plants

electrical grids

river networks

Motivation

Aim: optimize global objective, but

- keep subsystem models and their data locally
- decide as much as possible on local level
- distribute computations

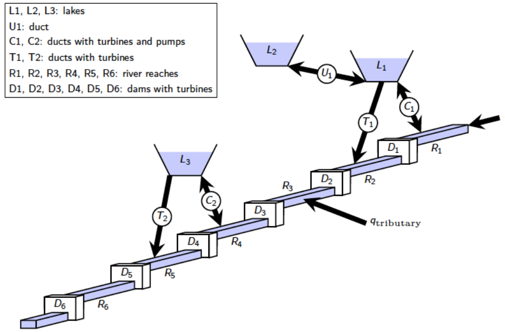
“Think globally, act locally”

Crucial Assumption

Local subsystem "simulation boxes" exist.

- use their own discretization scheme
- use their own modelling language
- solve local ODE, DAE, or PDE model
- **can generate derivatives (sensitivities)**

Benchmark Problem:
Hydro Power Valley (HPV)



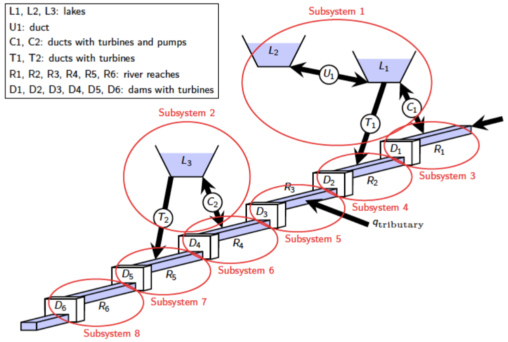
HD–MPC

SEVENTH FRAMEWORK PROGRAMME

Large scale model inspired by a real hydro power valley of Electricite de France [1]

[1] Savorgnan, Romani, Kozma, Diehl. *Journal of Process Control*, 21(5), 738-745, 2011

Benchmark Problem: Hydro Power Valley (HPV)



Large scale model inspired by a real hydro power valley of Electricite de France [1]

[1] Savorgnan, Romani, Kozma, Diehl. *Journal of Process Control*, 21(5), 738-745, 2011

PDE Model for one River Reach

Each reach modelled by 1D Saint-Venant equation:

$$\frac{\partial q(t,z)}{\partial z} + \frac{\partial s(t,z)}{\partial t} = 0$$

$$\frac{1}{g}\frac{\partial}{\partial t}\left(\frac{q(t,z)}{s(t,z)}\right) + \frac{1}{2g}\frac{\partial}{\partial z}\left(\frac{q^2(t,z)}{s^2(t,z)}\right) + \frac{\partial h(t,z)}{\partial z} + I_{\mathrm{f}}(t,z) - I_0(z) = 0$$

Discretize in space. Obtain ODE with 40 states, $x^i(t)$:

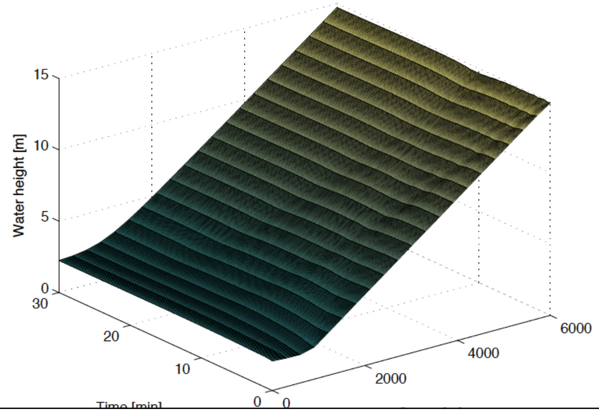
$$\dot{x}^i(t) = f^i(x^i(t), u^i(t), z^i(t))$$

Two types of inputs: controls $u^i(t)$ (turbine setting)

and coupling inputs $z^i(t)$ (inflow from above)

Simulation of one Reach

- use CVODES from Sundials Suite
- simulate reach for 30 minutes (constant $u^i(t)$)



Global Control Problem

- connect all 8 subsystems, regard 24 hours (= 48 time intervals)
- constrain water level variations
- two objectives: L1 to track power profile, L2 to track water levels

$$\min_{x_i,u_i} \int_0^T \gamma\,|e(t)|\,dt + \sum_{i=1}^{8} \int_0^T (x_i(t) - x_{ss,i})^T Q_i (x_i(t) - x_{ss,i}) dt$$

$$\text{s.t.} \quad \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \vdots \\ \dot{x}_8(t) \end{bmatrix} = f\left(\begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_8(t) \end{bmatrix}, \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_8(t) \end{bmatrix} \right)$$

$$(x_i(t), u_i(t)) \in C_i \quad i = 1, \ldots, 8$$

$$x_i(t) = x_{i,0} \quad i = 1, \ldots, 8$$

$$e(t) = p_r(t) - \sum_{i=1}^{8} p_i(x_i(t), u_i(t))$$

Overview

- Large Interconnected Systems
- **Distributed Multiple Shooting**
- Outlook: Distributed Quadratic Programming

Decomposable Formulation

Coupling inputs $z^i(t)$ obtained from neighbor's outputs $y^i(t)$

$$\begin{aligned}
\min_{\substack{x,u,z,\\ y,e}} \quad & \int_0^T \ell(e(t))dt + \sum_{i=1}^{M} \int_0^T \ell^i(x^i(t), u^i(t), z^i(t))dt \\
\text{s.t.} \quad & \dot{x}^i(t) = f^i(x^i(t), u^i(t), z^i(t)) \\
& y^i(t) = g^i(x^i(t), u^i(t), z^i(t)) \\
& x^i(0) = \bar{x}_0^i \\
& z^i(t) = \textstyle\sum_{j=1}^{M} A_{ij} y^j(t) \\
& e(t) = r(t) + \textstyle\sum_{i=1}^{M} B^i y^i(t) \\
& p^i(x^i(t), u^i(t)) \geq 0, \quad q(e(t)) \geq 0 \quad t \in [0, T]
\end{aligned}$$

Decomposable Formulation

Coupling inputs $z^i(t)$ obtained from neighbor's outputs $y^i(t)$

$$
\begin{aligned}
\min_{\substack{x,u,z,\\ y,e}} \quad & \int_0^T \ell(e(t))dt + \sum_{i=1}^{M} \int_0^T \ell^i(x^i(t), u^i(t), z^i(t))dt \\
\text{s.t.} \quad & \dot{x}^i(t) = f^i(x^i(t), u^i(t), z^i(t)) \\
& y^i(t) = g^i(x^i(t), u^i(t), z^i(t)) \\
& x^i(0) = \bar{x}_0^i \\
& z^i(t) = \textstyle\sum_{j=1}^{M} A_{ij} y^j(t) \\
& e(t) = r(t) + \textstyle\sum_{i=1}^{M} B^i y^i(t) \\
& p^i(x^i(t), u^i(t)) \geq 0, \quad q(e(t)) \geq 0 \quad t \in [0, T]
\end{aligned}
$$

Decomposable Formulation

Coupling inputs $z^i(t)$ obtained from neighbor's outputs $y^i(t)$

$$
\begin{aligned}
\min_{\substack{x,u,z,\\ y,e}} \quad & \int_0^T \ell(e(t))dt + \sum_{i=1}^{M} \int_0^T \ell^i(x^i(t), u^i(t), z^i(t))dt \\
\text{s.t.} \quad & \dot{x}^i(t) = f^i(x^i(t), u^i(t), z^i(t)) \\
& y^i(t) = g^i(x^i(t), u^i(t), z^i(t)) \\
& x^i(0) = x_0^i \\
& z^i(t) = \textstyle\sum_{j=1}^{M} A_{ij} y^j(t) \\
& e(t) = r(t) + \textstyle\sum_{i=1}^{M} B^i y^i(t) \\
& p^i(x^i(t), u^i(t)) \geq 0, \quad q(e(t)) \geq 0 \quad t \in [0, T]
\end{aligned}
$$

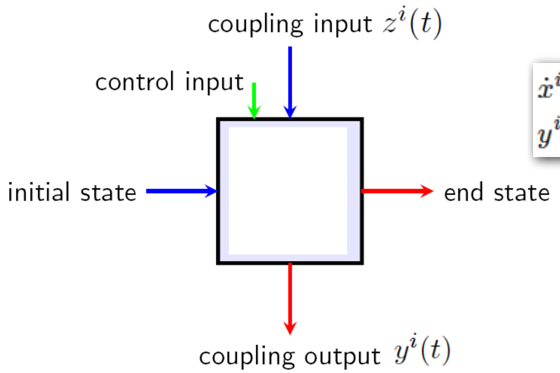
Decomposable Formulation

Coupling inputs $z^i(t)$ obtained from neighbor's outputs $y^i(t)$

$$\min_{\substack{x,u,z,\\ y,e}} \quad \int_0^T \ell(e(t))dt + \sum_{i=1}^{M} \int_0^T \ell^i(x^i(t), u^i(t), z^i(t))dt$$

$$\begin{aligned}
\text{s.t.} \quad & \dot{x}^i(t) = f^i(x^i(t), u^i(t), z^i(t)) \\
& y^i(t) = g^i(x^i(t), u^i(t), z^i(t)) \\
& x^i(0) = x_0^i \\
& z^i(t) = \textstyle\sum_{j=1}^M A_{ij} y^j(t) \\
& e(t) = r(t) + \textstyle\sum_{i=1}^M B^i y^i(t) \\
& p^i(x^i(t), u^i(t)) \geq 0, \quad q(e(t)) \geq 0 \quad t \in [0, T]
\end{aligned}$$

One Simulation Box

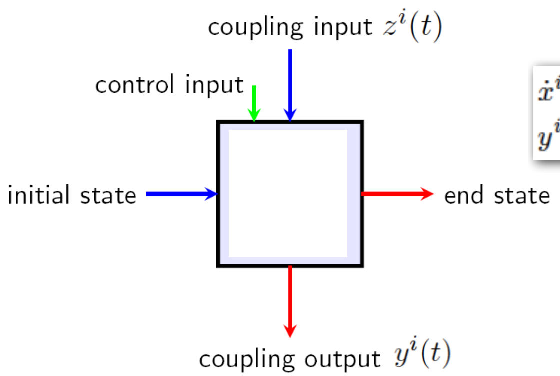


$$\dot{x}^i(t) = f^i(x^i(t), u^i(t), z^i(t))$$
$$y^i(t) = g^i(x^i(t), u^i(t), z^i(t))$$

$t \in [t_n, t_{n+1})$

Key idea: represent coupling variables $z^i(t)$, $y^i(t)$ by finite basis, e.g. orthogonal Legendre polynomials (of high degree)

One Simulation Box



$$\dot{x}^i(t) = f^i(x^i(t), u^i(t), z^i(t))$$
$$y^i(t) = g^i(x^i(t), u^i(t), z^i(t))$$

$$t \in [t_n, t_{n+1})$$

$$x^i_{n+1} = F^i_n(x^i_n, u^i_n, \mathbf{z}^i_n)$$
$$\mathbf{y}^i_n = G^i_n(x^i_n, u^i_n, \mathbf{z}^i_n)$$

Key idea: represent coupling variables $z^i(t)$, $y^i(t)$ by finite basis, e.g. orthogonal Legendre polynomials (of high degree)

Result: Decomposable NLP

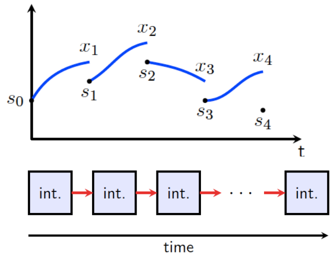
"Distributed Multiple Shooting"

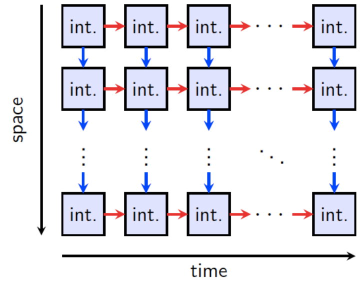
$$
\begin{aligned}
\min_{\substack{u_n^i, x_n^i, \mathbf{z}_n^i,\\ \mathbf{y}_n^i, \mathbf{e}_n}} \quad & \sum_{n=0}^{N-1} \left(L_n(\mathbf{e}_n) + \sum_{i=1}^{M} L_n^i(x_n^i, u_n^i, \mathbf{z}_n^i) \right) \\
\text{s.t.} \quad & x_{n+1}^i = F_n^i(x_n^i, u_n^i, \mathbf{z}_n^i) \qquad n = 0, \dots, N-1 \\
& \mathbf{y}_n^i = G_n^i(x_n^i, u_n^i, \mathbf{z}_n^i) \qquad n = 0, \dots, N-1 \\
& x_0^i = \bar{x}_0^i \\
& \mathbf{z}_n^i = \textstyle\sum_{i=1}^{M} A_{ij} \mathbf{y}_n^j \\
& \mathbf{e}_n = \mathbf{r}_n + \textstyle\sum_{i=1}^{M} B_{ij} \mathbf{y}_n^j \\
& p^i(x_n^i, u_n^i) \geq 0, \qquad Q_n(\mathbf{e}_n) \geq 0
\end{aligned}
$$

Here: M=8 systems, N=48 time intervals

Distributed Multiple Shooting

Standard MS [1] vs. Distributed MS [2]





Advantages: even more parallelism and sparsity.

[1] Bock, Plitt: A multiple shooting algorithm for direct solution of optimal control problems. *9th IFAC World Congress*, Budapest, 243–247, 1984.

[2] Savorgnan, Romani, Kozma, Diehl: Multiple shooting for distributed systems with applications in hydro electricity production. *Journal of Process Control*, 21(5), 738-745, 2011

Sequential Convex Programming (SCP)

Summarize problem: $\min_{x,u} f(x,u) \quad \text{s.t.} \quad \phi(x,u) = x, \quad (x,u) \in \Omega$

with f , Ω convex, ϕ nonlinear. SCP[2] generalizes SQP[1]. It solves in each iteration a convex subproblem:

$$\begin{aligned} \min_{x,u} \quad & f(x,u) \\ \text{s.t.} \quad & \phi(\bar{x},\bar{u}) + \frac{\partial \phi}{\partial x}(x-\bar{x}) + \frac{\partial \phi}{\partial u}(u-\bar{u}) = x \\ & (x,u) \in \Omega \end{aligned}$$

[1] Powell, M.: Algorithms for nonlinear constraints that use Lagrangian functions *Mathematical Programming*, 1978, 14, 224-248

[2] Tran Dinh, Savorgnan, Diehl: Adjoint-based predictor-corrector sequential convex programming for parametric nonlinear optimization. *SIAM Journal on Optimization* (in print)

SCP Implementation

Two main computational steps per SCP iteration:

- “simulation box” evaluations incl. derivatives (parallel)
- convex subproblems (CPLEX, parallel)

Used environment:

- 16 Core Workstation (2.7GHz Intel Xeron CPUs)
- written in C++, use openmp for parallelization
- CVODES from Sundials package for ODE sensitivities

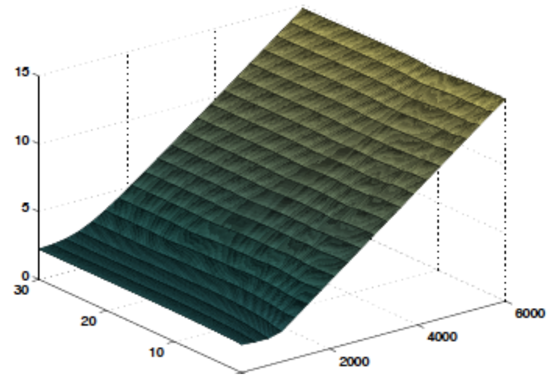
All written in CasADi optimization language

CasADi

- "Computer Algebra System for Automatic Differentiation"
- Implements AD on sparse matrix-valued computational graphs
- Open-source tool (LGPL): www.casadi.org, developed at OPTEC by Joel Andersson and Joris Gillis
- Front-ends to C++, Python and Octave
- Symbolic model import from Modelica (via Jmodelica.org)
- Interfaces to: SUNDIALS, CPLEX, qpOASES, IPOPT, KNITRO,
- "Write efficient optimal control solver in a few lines"

HPV Control Problem

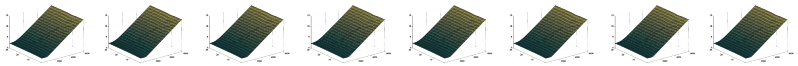
One simulation box = one reach on one interval



CPU time including all derivatives: 3 seconds

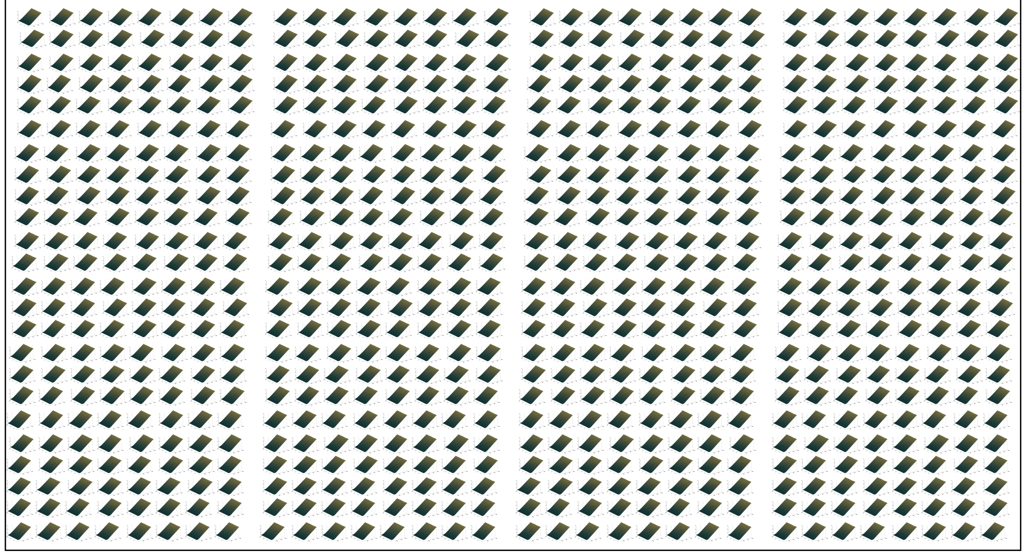
HPV Control Problem

8 simulation boxes = all systems on one interval



HPV Control Problem

8 x 48 = 384 simulation boxes



HPV Control Problem

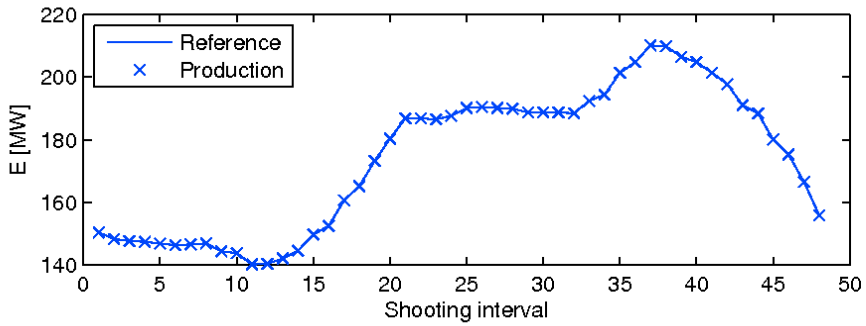
8 x 48 = 384 simulation boxes

Time per Iteration (on 16 cores): 180 sec

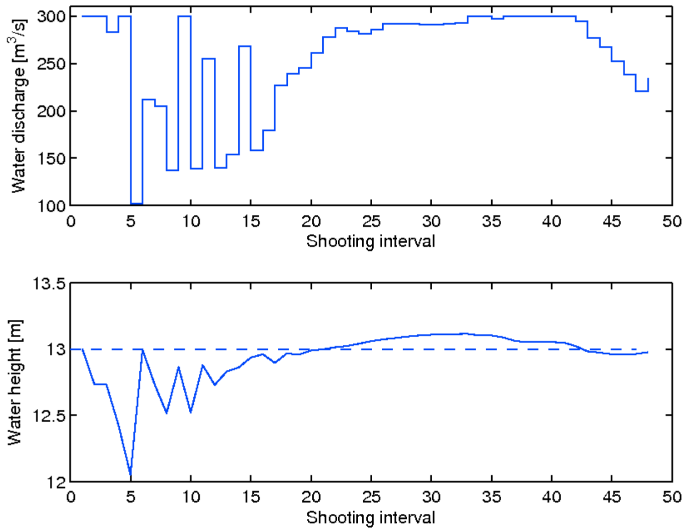
In case of infinitely many cores (est.):

- Distributed MS:	3+4 =	7 sec
- Standard MS:	34+4 =	38 sec
- Single Shooting:	1632+4 =	1636 sec

Result, after 11 SCP iterations: Power Tracking



Result: one control (of 12), one water level (of 120)



Overview

- Large Interconnected Systems
- Distributed Multiple Shooting
- **Outlook: Distributed Quadratic Programming**

Distributed Convex Optimization Algorithms

How to best solve convex problems in a distributed way?

- Dual Decomposition with Fast Gradient Schemes?
- Alternating Direction Method of Multipliers (ADMM)?
- Dual decomposition, smoothing and excessive gap? [1]

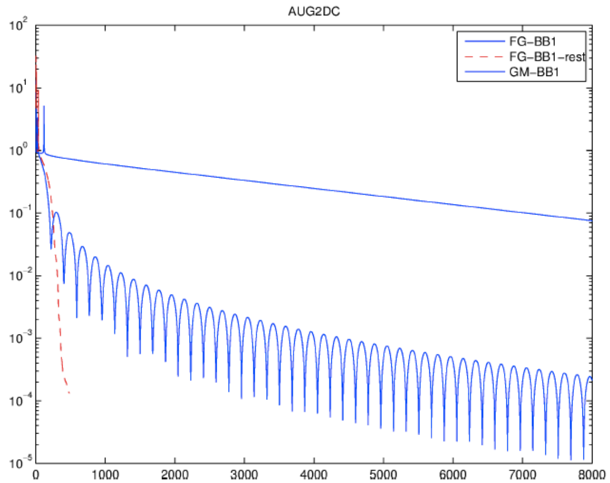
Problem of first order methods: sublinear convergence

[1] Tran Dinh, Savorgnan, Diehl: Combining Lagrangian Decomposition and Excessive Gap Smoothing Technique for Solving Large-Scale Separable Convex Optimization Problems, submitted to Computational Optimization and Applications (in review)

Sublinear convergence with very different speeds...

Preliminary results for 3 Lagrangian Decomposition schemes (on one test problem):

Accuracy vs. Iteration Count



Second Order Methods & Iterative Linear Solvers ?

- Sparse Decomposable Interior Point Methods? e.g. [1,2]
- Parallel Active Set Methods? Preliminary comparison from [3] :

Number of iterations	Example 1	Example 2
Parallel Active-Set Algorithm	7	9
Sparse Interior-Point Algorithm	12	6
Fast Gradient Method	>1000	>10000

[1] Jacek Gondzio and Andreas Grothey: Exploiting structure in parallel implementation of interior point methods for optimization. Comp. Man. Sc., Vol 6, No 2 (2009)
[2] Tran Dinh, Necoara, Savorgnan, Diehl: An inexact perturbed path-following method for Lagrangian decomposition in large-scale separable convex optimization. SIAM Opt. (in revision)
[3] Ferreau, Kozma, Diehl: A Parallel Active-Set Strategy to Solve Sparse Parametric Quadratic Programs Arising in MPC, Proceedings of NMPC 2012, Noordwijkerhout, 2012

Decomposable QP Benchmark Collection

Name	m	n	#eq	#ineq	conv.
AUG2DC	20200	5	10000	0	S
AUG2DCQP	20200	5	10000	40400	S
CONT-100	10197	2	9801	20394	S
CONT-200	40397	8	39601	80794	S
DTOC3	14999	5	9998	2	S
2nd-ord-ch	117760	128	79360	235520	S
HPV-full	27812	384	26976	2518	C
HPV-sys	27812	8	26976	2518	C
SMOKE	151250	201	75440	148091	C
AUG2D	20200	5	10000	0	C
AUG2DQP	20200	5	10000	20200	C
CONT-101	10197	2	10098	20394	C
CONT-201	40397	4	40198	80794	C
CONT-300	90597	9	90298	181194	C
UBH1	18009	18	12000	12030	C

(with C. Conte, M. Morari)

Consider to add your decomposable QPs to the benchmark collection

m: number of variables, n : number of subproblems

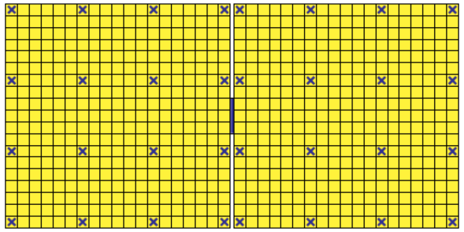
Conclusions and Future Work

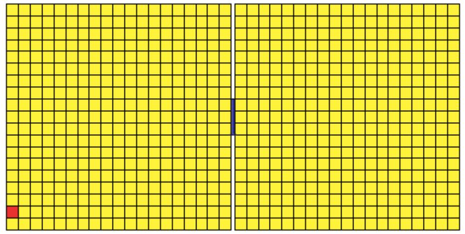
- Large systems with many subsystems and local decisions are a challenge for real-time optimization
- Distributed Multiple Shooting offers a way to keep models and most computations local - speed-up of 200 for HPV test problem possible
- Distributed convex optimization: need to compare first order methods, parallel interior point, and parallel active set methods

HPV System Dimensions

Subsystem #	x	u	z	y
1	2	3	12	3
2	1	3	11	3
3	41	1	7	11
4	41	1	11	16
5	41	1	11	11
6	41	1	11	16
7	41	1	11	16
8	41	1	6	6
$\sum$	249	12	80	82

Smoke Detection Problem



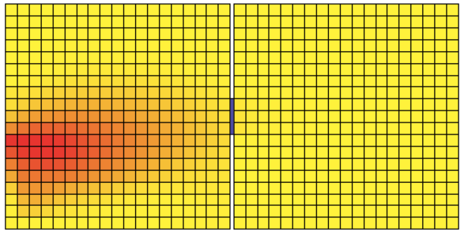


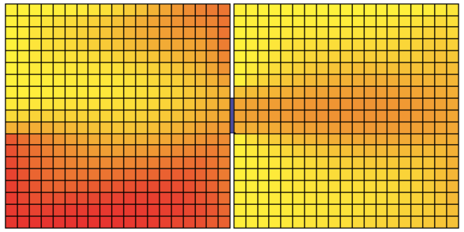
- Smoke sensors located in two connected rooms
- Aim: recover source location and time
- Source known to be sparse in time and space: use L2 fit with L1 regularization

Smoke Detection Problem

- Some pictures from simulation of PDE:

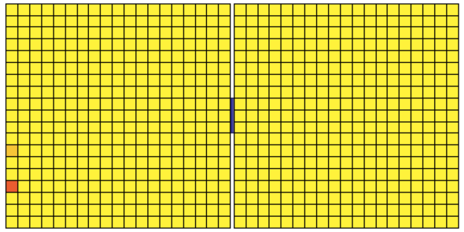
$$\frac{\partial \psi}{\partial t} = D\Delta\psi - v^f \nabla\psi + u(t,x) + h(\psi)$$





Smoke: Solution and Runtime Comparison

- NLP Solver correctly identifies source from 2^(8664) possibilities
- Distributed Multiple Shooting 10 x faster than single shooting
- Next bottleneck: QP



Method	Sens.	QP sol.	# opt. vars.
SS	452.4s	3.4s	8664
MS	79.8s	35.8s	15162
DMS	15.7s	23.8s	15342